APPENDIX 1

```
void sw ( )
(

#define iw = 12;                          /* instruction
                                          width */
#define mw = 3:                           /* memory width */
#define CONST = 0                         /* push constant */
#define LOAD = 1                          /* push variable */
#define GLOBAL = 2                         /* push address */
#define PUTCHAR, = 15 /*                  put a character along the
                                          standard output channel*/
#define GETCHAR = 16 /*                   get a character from the
                                          standard input channel */


...

rom program []
#include "prog.o" ): ram stack[1«mw] with dualport = 1 ];
ram memory[1«mw] unsigned iw PC, ir, tos;
unsigned mw sp;

do par it = program[pc]: PC = PC + 1;
tos = stack[sp-1];                           /* save top of
                                             stack to avoid
                                             two ram accesses
                                             in one cycle
                                             */


switch (ir)
case
CONST par
        stack[sp] = program[pc];
        sP = sP+l:
        PC = Pc+l:
        ]
        break;
case LOAD
        stack[sp-1] = memory[tos<-mw];
        break;
case STOP break; default :                   /* unknown opcode */
while (1) delay;

] while (ir != STOP);
```

]

Register transfer level description of simple processor

APPENDIX 2

```
void main() { char hwswchan;
char unsigned 8 port:

        par {
                parallel_,,port(port);
                SyncGen():

                        initialiseRam(port);
                        par {
                                display(hwswchan): sw(hwswchan);
                                y 1 }
        }

        RTL description of main
```

APPENDIX 3

CALCULATION PROCESS

```
/*
* Channel communicating object positions
*/ chap unsigned 17 position;

/*
* Channel communicating segment information
*/
chanout unsigned 9 segment;

/*
* Channel communicating button information
*/
chanin unsigned 2 buttons;

/*
* Overall par
*/ par

          /*
        * Mass motion
        */

            /*
            * Positions of each mass, 9+8 fixed point
              */
            unsigned 17 p0, pl, p2, p3, p4, p5, p6, p7;
            /*
            * Velocity of each mass, 9+8 fixed point
              */
            int 17 vl, v2, v3, v4, v5, v6, v7; '
            /*
            * Accelerations of each mass, 9+8 fixed point
              */
            int 17 al, a2, a3, a4, a5, a6, a7;
            /*
            * Sutton status
              */
            unsigned 2 button status;
            /*
            * Initial setup of positions
              */
```

```
p0 = 65536;
pl = 65536;
p2 = 65536;
p3 = 65536;
p4 = 65536;
p5 = 65536;
p6 = 65536
p7 =65536


 /*
 * Forever
 */
while (1)
   {

   /*
     * Send successive positions down position channel
     */
   send(position, p0);
   send(position, p1);
   send(position, pl);
   send(position, p2);
   send(position, p2);
   send(position, p3);
   send (position, p3);
   send(position, p4);
   send(position, p4);
   send(position, p5);
   send(position, p5);
   send(position, p6);
   send(position, p6);
   send(position, p7);

   /*
   * Update positions according to velocities
     */
   pl +_ (unsigned 17)vl;
   p2 +_ (unsigned 17)v2;
   p3 +_ (unsigned 17)v3;
   p4 +_ (unsigned 17)v4;
   p5 +_ (unsigned 17)v5;
   p6 +_ (unsigned 17)v6;
   p7 +_ (unsigned 17)v7;

     /*
```

```
         * Update velocities according to acceleratiors
         */
        vl += al - (v1 » 6);
        v2 += a2 - (v2 » 6) ;
        v3 += a3 - (v3 » 6);
        v4 += a4 - (v4 » 6);
        v5 += a5 - (v5' » 6);
        v6 += a6 - (v6 » 6);
        v7 += a7 - (v7 » 6);

        /*
         * Set accelerations according to relative positions
         */
        a1 = (int 17)(((p2 » 8) - (pl » 8)) + ((p0 » 8) - (pl » 8)));
        a2 = (int 17)(((p3 » 8) - (p2 » 8)) + ((pl » 8) - (p2 » 8))):
        a3 = (int 17)!!(p4 » 8) - (p3 » 8)) + ((p2 » 8) - !p3 » 8)));
        a4 = (int 17)(((p5 » 8) - (p4 » 8)) + ((p3 » 8) - (p4 » e> >;
        a5 = (int 17)((!p6 » 8) - (p5 » 8)) + ((p4 » 8) - (p5 » 8)));
        a6 = (int 17)(((p7 » 8) - (p6 » 8)) + ((p5 » 8) - (p6 » e > ) ;
        a7 = (int 17)((p6 » 8) - (p7 » 8));

        /*
         * Get button information
         */
        receive(buttons, button status);

         /*
         * Fix top point according to buttons
         */ if (button status & 1)

          p0 = 65536 - 16384;
          )
  else    if (button status & 2)
          (
            p0 = 65536 + 16384;

  else

            p0 = 65536;
          }
   )
   /*
     * nine drawing
     */
   (
     /*
     * Positions of previous and next massess positions
     */
    unsigned 17 prev_.pos, next pos, curr pos;
    /*
     * Which line of interpolation
     */
    unsigned char line;
    /*
     * Forever
     */
    while (1)
       (
       /*
         * Receive previous mass position
         */
       receive (position, prev posy;
       curr pos = prev pos;
       /*
         * Read next mass position
         */
       receive(position, next posy;
       /*
         * Do 64 lines of interpolation
         */
       for (line = 0; line != 64; line++)
          (
             /*
               * Send start position of segment
               */
             send(segment, curr pos » 8);      /**width adjustment:17 along
                                                 channel of width 9 so takes bottom 9
                                                 bits*/

             /*
               * Move by appropriate amount (1/64  total change)
               */
             curr pos +_ (unsigned 17)(((int 17)next pos -
                                          (int 17)prev pos) » 6);
             /*
               * Send end position of segment
               */
             send(segment, curr pos » 8):
          )
       )
 )
)
```

DISPLAY PROCESS

```
/* standard includes */
                    #include "hammond.h"
          #include "syncgen.h"
          #include "stdlib.h"
          #include "parallel.h"

                /*
* Segment information channel */ chap segment;

                /*
* Button information channel */
chan buttons:

                /
* Include dash generated stuff */
#include "handelc.h"

                /*
* Main program */
void main() (
               /
* Scan positions
*/ unsigned sx, sy;

               /
* Vdeo output register
*/
unsigned 1 video;

               /*
* Video output bus
*/

interface bus out() video out(Visible(sx, sy) ?
(video ? (unsigned 12)Oxfff : 0) 0) with video spec;

#ifndef SIMULATE
               /*
* Left button input bus
*/
interface bus in (unsigned 1) button_left()
```

```
                with button white spec;

        /*
        * Right button input bus
        */
                interface bus in(unsigned 1) button right()
                    with button_black spec;
            #endif

        /*
        *
                      Overall par
        */ par {
        /*
                        * VGA sync generator
                        */
                      SyncGen(sx, sy, hsync pin, vsync pin);
                      /*
                        *
                          Dash generated hardware
                        */
                      hardware();
                      /*
                        * Run-length decoder
                        */
                      {
                      /*
                        * Segment start and end positions
                        * /
                      unsigned start, end;
                      /*
                        * Forever
                        */
                      while (1)
                          {
                            while (sy != 448)
                              /*
                                * Read segment information
                                */
                              segment ? start;
                              segment ? end;
                              /*
                                * Get in the right order
                                */
                              if (start > end)
                                  {
```

```
                        par
                            {

end = start;
start = end;.
)


                    /*
* Make at least 1 pixel visible
*/
if (start == end)
                        end++;

/*
                        * Wait
*/
                    while (sx != 0)
                        delay;
                    /*
                      * Draw a scanline worth
                      * /
                    while (sx != 512)
                          if ((sx <- 9) >= start && (sx <- 9) < end)

                              video = 1;
                          else
                        video = 0;

                        )
                  /*
                    * Communicate button status
                    */
        #ifdef SIMULATE
                  buttons ! 1;
        #else
                    buttons ! button left.in @ button right.in;
        #endif
                  /*
                    * Wait
                        */
                  while (sy != 0)
                      delay;
                )
```

United States Patent & Trademark Office

Office of Initial Patent Examination

Application papers not suitable for publication

SN 10026312 Mail Date 01/07/02

☐ Non-English Specification

☐ Specification contains drawing(s) on page(s)________ or table(s)________

☐ Landscape orientation of text ☐ Specification ☐ Claims ☐ Abstract

☐ Handwritten ☐ Specification ☐ Claims ☐ Abstract

☐ More than one column ☐ Specification ☐ Claims ☐ Abstract

☐ Improper line spacing ☐ Specification ☐ Claims ☐ Abstract

☐ Claims not on separate page(s)

■ Abstract not on separate page(s)

☐ Improper paper size -- Must be either A4 (21 cm x 29.7 cm) or 8-1/2"x 11"

- ☐ Specification page(s)________ ☐ Abstract
- ☐ Drawing page(s)________ ☐ Claim(s)

☐ Improper margins

- ☐ Specification page(s)________ ☐ Abstract
- ☐ Drawing page(s)________ ☐ Claim(s)

☐ Not reproducible

Reason	Section
☐ Paper too thin	☐ Specification page(s)________
☐ Glossy pages	☐ Drawing page(s)________
☐ Non-white background	☐ Abstract
	☐ Claim(s)

☐ Drawing objection(s)

- ☐ Missing lead lines, drawing(s)________
- ☐ Line quality is too light, drawing(s)________
- ☐ More than 1 drawing and not numbered correctly
- ☐ Non-English text, drawing(s)________
- ☐ Excessive text, drawing(s)________
- ☐ Photographs capable of illustration, drawing(s)________

10026312.122101